

Dijkstra Worksheet

Name:

NetID:

Signature:

1. What is the defining characteristic of a Max Heap? How can the characteristics of Heaps be used to implement a Priority Queue?

Ans: Parents must have larger values than their children, and the root node has the biggest value. In a Min Heap, parents must always have a smaller value than the child, so the root node is always the node with the smallest value.

2. Suppose we have an array representation of a Binary Heap. Find the parent, left child, and right child of an element i .

Ans: Parent $\rightarrow \text{Floor}((i-1)/2)$

Left Child $\rightarrow 2*i+1$

Right Child $\rightarrow 2*i+2$

3. How can you insert into a Heap? Assume the Heap is implemented as an array. Write some code!

```
1 def HeapInsert(Heap, key):
2     Heap.size = Heap.size + 1
3     i = Heap.size
4     while ((i > 0) and (Heap[parent(i)] > key)):
5         Heap[i] = Heap[parent(i)]
6         i = parent(i)
7
8     Heap[i] = key
```

4. Let's say you have a very important packet to send from your computer to another computer, find the shortest path from your computer to all other nodes in the computer network.

Here is Dijkstra:

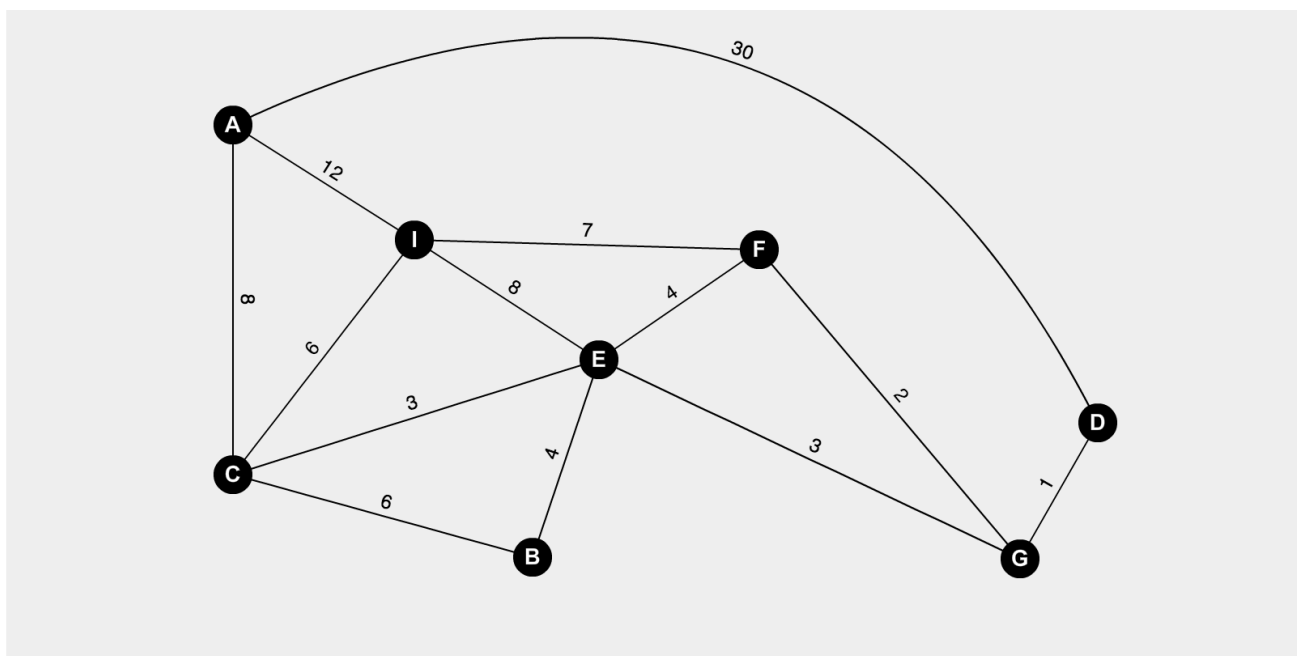
```
1 def Dijkstra(Graph, start, end):
2     PQ = Queue.PriorityQueue()
```

```

3  foreach node in Graph:
4      node.cost = infinity
5  start.cost = 0
6  foreach node in Graph:
7      PQ.insert(n,node.cost)
8  while not(PQ.empty()):
9      u = PQ.extract()
10     if u == end:
11         break
12     foreach neighbor v in u.neighbors:
13         w = cost from v to u
14         newCost = u.cost + w
15         if (newCost < v.cost):
16             PQ.decreasekey(v,newcost)
17             v.cost = newCost
18             v.predecessor = u
19
20

```

Here is a graph. Use Dijkstra to find the shortest path from A to all nodes in the graph. Go through iteration by iteration, update the costs, as well as u from the psuedocode. ∞



	<i>u</i>	A	B	C	D	E	F	G	I
1	A	0	inf	8	30	inf	inf	inf	12
2	AC	0	14	8	30	11	inf	inf	12
3	ACE	0	14	8	30	11	15	14	12
4	ACEI	0	14	8	30	11	15	14	12
5	ACEIG	0	14	8	15	11	15	14	12
6	ACEIGB	0	14	8	15	11	15	14	12